8-2020

Privacy preserving search services against online attack

Yi ZHAO

Jianting NIAN

Kaitai LIANG

Yanqi ZHAO

Liqun CHEN

See next page for additional authors

Author

Yi ZHAO, Jianting NIAN, Kaitai LIANG, Yanqi ZHAO, Liqun CHEN, and Bo YANG

Privacy preserving search services against online attack

Yi Zhao[a,b,c] , Jianting Ning[d,e], Kaitai Liang[f] Yanqi Zhao[b,c], Liqun Chen[f], Bo Yang [b,c,*]

a School of Information Engineering, Chang'an University, Xi'an 710064, China
b School of Computer Science, Shaanxi Normal University, Xi'an 710062, China
c State Key Laboratory of Information Security Institute of Information Engineering, Chinese Academy of Sciences, Beijing 10 0 093, China
d Department of Computer Science, Fujian Normal University, China
e Department of Computer Science, Singapore Management University, Singapore
f Department of Computer Science, University of Surrey, UK

Published in Computers and Security, August 2020, 95, 101836, pp. 1-10. DOI: 10.1016/j.cose.2020.101836

Abstract: Searchable functionality is provided in many online services such as mail services or outsourced data storage. To protect users privacy, data in these services is usually stored after being encrypted using searchable encryption. This enables the data user to securely search encrypted data from a remote server without leaking data and query information. Public key encryption with keyword search is one of the research branches of searchable encryption; this provides privacy-preserving searchable functionality for applications such as encrypted email systems. However, it has an inherent vulnerability in that the information of a query may be leaked using a keyword guessing attack. Most of existing works aim to make the system resistant to offline keyword guessing, but this does not protect against online attacks on real world services. In this paper, we move a step forward to present a generic framework able to resist online keyword guessing attack using a server-assisted model. Specifically, we design a novel primitive C mirrored all-but-one lossy encryption, which can prevent a specific user from generating valid encryptions. This primitive can be seen as an access control on encryption ability. Combining searchable encryption technique with the new primitive makes online keyword guessing attack impossible for the specified user, even if the attack is launched online. We further give formal security analysis for the generic framework, and a concrete implementation with efficiency analysis to show that our design is practical.

Keywords: Keyword search, Encrypted data, Security, Online keyword guessing attack

1. Introduction

Searchable functionality is popular in many online services like email system and data storage. One can search his email box with date to find the emails he received at the date. Or one can also use keywords to find out those goods with names which contain the keywords in online shopping. However, searching by keywords makes a threat to user's privacy. For example, attackers can use data mining technology to analyse the search history of an online merchant user to determine his hobby or income range. So in practice, some cryptographic tools are used in the protocol to protect privacy, like encrypted email systems.

Encryption technologies (e.g., AES(Advanced Encryption Standard), DES (Data Encryption Standard), RSA (Rivest et al., 1978) have been widely employed in real-world applications, such as ProtonMail[1] and SpiderOak[2], to protect the confidentiality of client data. However, the encryption may prevent further operations over encrypted data. For instance, making use of RSA to encrypt emails and further store the encryptions to a remote server that makes search over data somewhat impossible on server side. This is because server may not be shared decryption rights by clients. There are some naive approaches to enable server to search over encrypted emails. One solution is to persuade clients to share secret keys with server, so that server can first decrypt and further search over plain data. This, however, may infringe the privacy of clients and meanwhile, it may not be necessary for clients to share fully trust with server. Another solution could be that clients downloads all encrypted emails to local and further fulfil throughout search over decryption. Although leaking no email information to server, this solution may yield heavy bandwidth and computational cost

[1] https://protonmail.com/.

[2] https://spideroak.com

to clients (for example, a client only would like to search one file but needs to download the whole encrypted email database).

Searchable encryption (SE) is introduced to tackle the above dilemma in the sense that a client only needs to send a search token (embedded with some keyword) to server so that the server is able to fulfil search over encrypted data and return the corresponding matching encrypted data without accessing underlying data and query content. Symmetric searchable encryption (SSE) (e.g., Du et al., 2018; Song et al., 2000) and public key encryption with keyword search (PEKS) (e.g., Boneh et al., 2004) are the two main search branches of SE so far. This paper will deal with the latter.

The first notion of PEKS is introduced in Boneh et al. (2004). PEKS is a general implementation of SE in public key context without using symmetric key distribution. This mechanism has attracted increasing attention in academic research and industry because it can provide versatile search functionalities for different security requirements in cloud storage services, such as mobile cloud applications (Liu et al., 2015) and electronic health record applications (Gritti et al., 2014). Under the framework of PEKS, a data sender usually encrypts files under receiver's public key and prepares an encrypted index structure corresponding to the encrypted files, and further sends both to a server. While needing to search and download files, the receiver generates a search token (hereafter, we refer to search token as search trapdoor) of some keyword and sends it to the server. With the trapdoor, the server makes some computation for searching and returns the matching (encrypted) results.

In general, a PEKS scheme can be implemented by an anonymous identity based encryption (IBE) Boneh et al. (2004). Keyword takes the role of identity and its corresponding secret key can be seen as a search trapdoor. When a trapdoor can be used to obtain a valid plaintext from the encryption under a keyword, the search is completed and the corresponding encrypted message is sent to the user. Since the construction is based on anonymous IBE, the keyword embedded into the search token will not be revealed by the server. In PEKS, the security concern is mainly about the leakage of query keyword which may result in information leak of search/access pattern and even the content of file. As pointed in Abdalla et al. (2008), the construction of PEKS requires the underlying IBE to be anonymous so that adversary can not determine the underlying keyword of a given encrypted file. However, Byun et al. (2006) found that given a trapdoor, an adversary can launch efficient attacks just by guessing and testing a search because it can encrypt "dummy" files from public key and any keyword as will. Therefore, the original scheme in Boneh et al. (2004) needs a secure channel to prevent outside attackers from launching keyword guessing attacks. To solve this problem, Rhee et al. (2009) and Fang et al. (2013) proposed schemes without leveraging secure channel. They put server's public key in the encryption algorithm so that only the one with corresponding secret key can fulfil a correct search test. This designated scheme is called secure channel free (SCF) because the trapdoor can be transmitted in public channel.

Many practical cases may need stronger security requirements than that of only achieving security against outside attackers. Cloud based service clients usually don't want to make others (including server) be able to infringe their privacy and data security. And moreover, it is indeed that service provider can cause damage much severer than outside attackers. But the vulnerability from server's keyword guessing attack (KGA) seems inherent in original framework of PEKS because server is supposed to have trapdoor (given by clients) and (self-generated) encrypted files, simultaneously, which can be seen as the "knowledge database". Therefore, server itself can definitely target some keywords to fulfil search test over the knowledge database. For example, an email server can send encrypted junk mails to a specified system user and use all search trapdoors (given by the email user) to search over the junk mails, which can help the server detect whether a keyword is embedded in those trapdoors.

A natural approach to solve this problem is making the keyword "fuzzy". Xu et al. (2013) proposed a fuzzy keyword search scheme in the sense that the server can only return a group of encrypted files so that it can not tell exactly which one is exactly needed by data user. However, this solution leaks the information of the range so that it is not formally secure against KGA. In fact if the set is relatively small, the server can still be able to find the right encrypted file with high probability. In another direction, Chen et al. (2016b) presented a novel framework in which the search testing process was handled by two servers without collusion. Soon Chen et al. (2016a) tried another approach to add an aided server outside the original framework because it is more convenient to handle in a blackbox way. These two schemes can successfully resist a weaker version of KGA, called off-line keyword guessing attack. This type of attack allows the server to attack by its own in off-line status after receiving considerable amount of trapdoors. In these two schemes, main server can't do search testing in an off-line status without the help of assisted server. But if the two server collude together (e.g., the two server share information), the keyword guessing attack still exists.

To see it in details, in Chen et al. (2016a) authors add an aided server and require both the sender and user to run a blind signature protocol with it. So without the knowledge of the secret injected by aided server, the main server can only test but not obtain any information of keyword. This result requires that the attack happens when aided server is off-line so the main server can not do the same thing as users. In practice, the time of attack is hard to anticipate so asking aided server to work off-line to avoid communication with main server seems impossible. Although they showed some mechanism to slow down the online key word guessing attack, this scheme does not rule out this type of attack because main server can query the aided server to inject the same secret to its guessing keyword and then test. So we can see that separating tester and encrypter is the way to resist online attack. That means test server are unable to generate PEKS ciphertext but only does the test.

1.1. Our contribution

We find that it can go further from existing two independent servers framework to resist stronger and online type of keyword guessing attack without adding new participants. Our scheme employs a new primitive to rule out "honest-registration online attack" in the framework of Chen et al. (2016a). In this kind of attack main server can query the aided server online even in malicious way like impersonating existing users, but it won't register valid dummy users. This model has practical value because most real applications today require registration with real identity verification. So registering dummy users to launch attack is not efficient and easily traceable. Moreover, we show how to adapt our scheme to handle with attacks allowing dummy users to most extent. Our main contributions can be listed as follows:

- Combining the notions of lossy encryption and key exchange protocol, we propose a novel primitive called mirrored all-but-one (ABO) lossy encryption. "mirrored" here means the security of this primitive is just the same as normal ABO lossy encryption schemes, except that while in normal case lossy branch is hidden to adversary and evaluation keys is public, this primitive uses the lossy branch known by adversary but makes computations on branches which is hidden. A special property of this scheme is that the "one" is even deprived of the capability to encrypt.

Table 1
Security comparison.

scheme	CKA	Offline KGA	Online KGA
Boneh et al. (2004)	√	×	×
Xu et al. (2013)	√	Partial	Partial
Chen et al. (2016a)	√	√	×
Ours	√	√	√

- With the new primitive, we propose our new PEKS scheme. Users are asked to encrypt messages to aided server through this scheme instead of sending directly. All valid users can run this protocol with aided server to obtain valid signature but only when the main server communicates with aided server it doesn't work. Even if the server sends the message on other valid users' behalf, the server still can't obtain any valid signature and the process is oblivious to the aided server. We formalize the security model and give generic construction with proof for this primitive. By combining classic blind signature scheme in Chen et al. (2016a) and ABO lossy encryption to build an ABO blind signature scheme, we can obtain a PEKS scheme secure against stronger type of keyword guessing attack. The security improvement can be seen in Table 1.

1.2. Related works

Other Type of Searchable Encryption. Apart from the aforementioned PEKS systems, there have been some variants of PEKS in the literature. Bellare et al. (2007) made use of deterministic encryption as a tradeoff between security and efficiency. Wang et al. Wang et al. (2007) leveraged dynamic accumulators to support the searchability in the context of multiple readers and writers. Hwang and Lee (2007) proposed a PEKS with conjunctive keyword search supporting multiuser. He et al. (2018) presented certificateless PEKS for internet of things context. Some research works have extended PEKS in the attribute-based setting to enhance search expressiveness, e.g., attribute-based searchable encryption (Han et al., 2018; Meng et al., 2017), regular language search (Liang et al., 2016a) and attribute-based search with sharing mechanisms (Liang et al., 2016b; Liang and Susilo, 2015). To see how attacks work on SE schemes, we refer to readers that Ning et al. (2019) investigates the vulnerability of SSE schemes based on the assumptions on how much prior knowledge an attacker is allowed to achieve and further presented a series of passive attacks. Besides (Du et al., 2018) considers forward privacy in dynamic updatable SSE. However, the problem that resisting online attack from main server who stores the ciphertext in PEKS is still open.

Lossy primitives. We employ the notion of 'lossy' cryptographic primitives in our construction. The idea of lossy primitives was originated from dual mode encryption in Peikert et al. (2008). This type of primitives usually have two working modes which can not be distinguished by attackers. The injective mode is used in normal setting and lossy mode is used in proof. This trick provides novel approaches for many problems. To find minimal assumptions for public key encryption (PKE) secure against chosen ciphertext attack (CCA), Peikert and Waters proposed the notion of lossy trapdoor functions (LTFs) (Peikert and Waters, 2011). To address the problem of PKE secure against selective opening chosen cipertext attack (SO-CCA), Bellare et al. defined lossy encryption and gave constructions from LTFs (Bellare et al., 2009). There are many variants of LTFs with different number of lossy branches, like all-but-one (ABO) LTFs (Peikert and Waters, 2011), all-but-N (ABN) LTFs (Hemenway et al., 2011) and all-but-many (ABM) LTFs (Hofheinz, 2012). In this paper, we will employ corresponding invariants in lossy encryption. Specifically, we will combine key exchange protocol and ABO lossy encryption with blind signature to provide access control for PEKS to rule out just main server.

1.3. Organization

The rest part of this paper is organized as follows: In Section 2 we introduce preliminaries needed. Then we present mirrored ABO lossy encryption protocol with security model and proof in Section 3. With the primitive we build two-server framework PEKS secure against online keyword guessing attack in Section 4. The remaining sections are performance and conclusion.

2. Preliminaries

In this section we introduce some knowledge which will be used in following sections.

2.1. Lossy primitives

2.1.1. (ABO) Lossy trapdoor functions

A collection of LTFs is a collection of publicly computable functions which are indexed by a set of public key $\{s\}$. Every public key is associated with a branch which is used to generate the key. There are two kinds of public keys, injective and lossy. Functions indexed by injective ones can be reversed to find pre-image. Functions indexed by lossy keys have smaller size of image than that of domain. We called the branch according to the former "injective branch" and the other "lossy branch". "lossy" means the image of the function working on these branches loses part of the information of the pre-image. We use a generalized notion to incorporate exponential lossy branches. Let $\{B_n\}$ denote a collection of branch sets and $\{B_n^*\}$ denote the corresponding collection of lossy branch sets. We recall the definition of ABO-LTFs Peikert and Waters (2011) below. If $\{B_n\}$ contains two elements only, it is just the standard LTF.

A collection of (n, k) ABO-LTFs is composed of 3 probabilistic polynomial time (PPT) algorithms:

G_{abo}: Take $\lambda \in \mathbb{N}$ and $b^* \in B_\lambda$ as input, output (s, td, B_λ^*), where s is a function index, td is its trapdoor and B_λ^* is the set of lossy branches that $b^* \in B_\lambda^*$.

F_{abo} and F_{abo}^{-1}: For any $b \in B_\lambda \setminus B_\lambda^*$, $F_{abo}(s, b, \cdot)$ computes an injective function $f_{s,b}(\cdot)$ over the domain $\{0,1\}^n$, and $F_{abo}^{-1}(s, td, b, \cdot)$ computes $f_{s,b}^{-1}(td, \cdot)$. For any $b \in B_\lambda^*$, $F_{abo}(s, b, \cdot)$ computes a function $f_{s,b}(\cdot)$ over the domain $\{0,1\}^n$ whose image size is at most 2^k.

There are two security requirements for ABO-LTFs:

Index indistinguishability: The ensemble $s \leftarrow G_{abo}(\lambda, b_0^*)$ and $s \leftarrow G_{abo}(\lambda, b_1^*)$ are computationally indistinguishable.

Lossy branch hidden: Any PPT adversary $\mathcal{A}$ who takes (s, b) as input, where $(s, td, B^*) \leftarrow G_{abo}(\lambda, b^*)$, has only a negligible probability to find a lossy branch b' such that $b' \neq b$ and $b' \in B^*$. And even $b \in B^*$, the probability is still negligible.

The above two properties can be used in the security proof. Usually the initial injective experiment can be changed to a lossy experiment via indistinguishability. And then using the second property to draw conclusions. In this paper, we will use lossy mode directly in practice.

2.1.2. Lossy encryption

Similar to LTFs, lossy public key encryption (LPKE) is a kind of PKE that has two working modes decided by its public key. The definitional difference between normal PKE and lossy one is just the key generation algorithm. Naturally, LPKE can be instantiated by LTFs.

A lossy encryption scheme is composed of 3 algorithms (G, E, D) satisfying:

1. Key generation algorithm G can generate two kinds of keys. $G(1^\lambda, inj)$ generates injective keys and $G(1^\lambda, lossy)$ generates lossy keys.
2. Under injective keys, encryption algorithm E and decryption algorithm D can work correctly. That is $\Pr[D(sk, E(pk, x; r)) = x] = 1$ where r is a random factor.

Under lossy keys, the generated ciphertext can't be properly decrypted. The schematic diagram is as follows.

$$G(1^\lambda, \cdot) = \begin{cases} pk_{inj}, E(pk_{inj}, m) = C_{inj}, D(sk, E(pk_{inj}, C_{inj})) = m \\ pk_{loss}, E(pk_{loss}, m) = C_{loss}, D(sk, E(pk_{loss}, C_{loss})) = \perp \end{cases}.$$

LPKE has the same secure property as LTFs. The two kinds of keys should be indistinguishable and information of plain text would be lost under lossy keys. Lossy branch hidden property is also implied, even if we extend this notion to ABO setting.

A simple construction of ABO lossy encryption from ABO LTFs and pair-wise independent hash function H is as follows:

- G: Run G_{abo} with b^* to obtain (s, td, B^*_λ). The public key is s.
- E: The ciphertext is computed as $c = (c_1, c_2) = ((F_{abo}(s, td, b, x), b), H(x) \oplus m)$ where x and b are chosen randomly, and m is the message.
- D: Given a ciphertext C, if $b \neq b^*$ compute $d = F^{-1}_{abo}(s, b, c_1)$, then $m = c_2 \oplus d$.

2.2. Non-interactive key exchange

Non-interactive key exchange (NIKE) is a collection of 3 algorithms (*Setup, Keygen, Sharedkey*) as follows.

- *Setup*: Generate a set of public parameters which both parties agree on.
- *Keygen*: Each party uses this algorithm to generate their own key pair (*pk, sk*).
- *Sharedkey*: On input one party's public key and the other party's secret key, output the shared key. This algorithm is commutative that we can change the party's role and the result remains the same. Specifically, given (ID_1, pk_1, sk_1) and (ID_2, pk_2, sk_2), $Sharedkey(ID_1, ID_2, pk_1, sk_2) = Sharedkey(ID_2, ID_1, pk_2, sk_1)$.

Security model: Usually a NIKE scheme is proved secure in CKS model which is proposed by Cash et al. (2009). This model allows adversary to query honestly generated public keys, shared key between existing honest users, associate public keys with other IDs or register dishonest users. With these abilities, the adversary still can not distinguish the shared key with randomness. We first focus on a widely implemented situation that CA needs verification with real identities (by text codes to cell phone etc.) for user registration so that registering dishonest users is not efficient. In this weaker case, Diffie–Hellman protocol is sufficient for implementation. Then we will show how to handle the case in which the adversary can register dummy users. In this case, we can adopt the NIKE scheme in Cash et al. (2009).

2.3. Blind signature

A blind signature scheme contains 5 algorithms including (*BS.gen, BS.blind, BS.sign, BS.unblind, BS.ver*):

- *BS.gen*: On input security number, signer runs this algorithm to generate verification key *vk* and signing key sk_s.
- *BS.blind*: On input a message m, output a blinded message m'. User sends m' to the signer.
- *BS.sign*: On receiving m', signer invokes signing algorithm to output a blinded signature s'.
- *BS.unblind*: On input a blinded signature s', output an original signature s.
- *BS.ver*: On input (*vk, m, s*), output 1 or 0.

A blind signature scheme should satisfy two security requirement.

- Blindness: The signer are unable to link any user with certain signed message. Formally, after two users ID_0 and ID_1 run the protocol with signer to obtain message-signature pair (m_0, s_0) and (m_1, s_1) respectively. Then choose $b \in \{0, 1\}$ randomly and send (m_b, s_b) to the signer, the signer outputs a guess b'. Define the advantage $Adv_{blind} = |\Pr[b' = b] - 1/2|$. The advantage is negligible against adversarial signer.
- Unforgeability: This is the same as normal security requirement of a signature scheme.

2.4. Public key encryption with key word search

Public key encryption with key word search (PEKS) is a collection of 4 algorithms as follows: A lossy encryption scheme is composed of 6 algorithms ($Kgen_R$, *PEKS, Trapdoor, Test*) as follows:

- $Kgen_R(\lambda)$: Receivers run this algorithm to generate public/secret key pair (pk_R, sk_R).
- $PEKS(pk_R, w)$: The server receives PEKS ciphertext CT_w generated by this algorithm.
- $Trapdoor(sk_R, w)$: Receivers generate trapdoor T_w by this algorithm.
- $Test(CT_w, T_w)$: Server runs this algorithm to test whether the ciphertext is the one receiver is searching for. Let the output be true/false.

Security definition:

The standard security requirement is that a PEKS scheme should be secure against keyword chosen attack (KCA). Our aim is to resist keyword guessing attack and this primitive is one building block so we omit the details of traditional CKA security here.

3. Mirrored ABO lossy encryption

We present this novel primitive to control the capability of encrypting messages in public key setting. This can be used to separate the role of encrypter and tester for main server in SA-PEKS.

3.1. Definition and security model

A mirrored ABO lossy encryption scheme is composed of 6 algorithms (*PP, Bgen, Brec, Gen, Enc, Dec*) as follows:

- $PP(\lambda)$: On input security parameters, output public parameters which all parties agree on, including the basic information like the group, hash function and index IND_i for every user.
- $Bgen(\lambda, IND_i)$: Every user indexed by IND_i runs this algorithm to generate a branch b_i for encryption between the receiver and itself.
- $Brec(IND_i)$: The receiver can recover the branch b_i used in encryption according to sender IND_i.
- $Gen(\lambda, IND^*)$: On input parameter λ and an chosen IND^*, output key pair (*lpk, lsk*) for receiver. The corresponding branch for IND^* is b^*.
- $Enc(lpk, IND_i, m)$: On input a branch b_i corresponding IND_i and a message m, output ciphertext (C, IND_i) where IND_i is the sender's index.
- $Dec(lsk, C, IND_i, b^*)$: First compute $b' = Brec(IND_i)$. If $b' = b^*$, output *reject*; Otherwise, compute $m = Dec(lsk, C, IND)$.

Secure against chosen plaintext attack (CPA): In injective mode, the encryption satisfies standard CPA security notion against every user.

- Lossiness: Given $Enc(lpk, b^*, m)$, m has large entropy that any adversary can recover the original m with only negligible probability.
- Branch Hidden: For $m \in \mathcal{M}$, $\{C = Enc(lpk, b, m); b \leftarrow Bgen(\lambda, IND)\} =_c \{C = Enc(lpk, b, m); b \leftarrow Rand\}$.
- Incapability of encryption: For the user ID^* associated branch b^*, it can generate a valid encryption only with negligible probability. We define this property in the game between a challenger and an adversary below:
  - Init: The adversary with IND^* runs $Bgen(\lambda, IND^*)$ to obtain its own branch b^*. The challenger computes $b^* = Brec(IND^*)$ and runs $Gen(\lambda, b^*)$ to generate (lpk, lsk). Then send lpk to the adversary.
  - Query: The adversary can query any IND's valid ciphertext without the knowledge of message.
  - Forge: The adversary outputs a tuple (m, C, IND). If the adversary can output a valid ciphertext such that $m = Dec(lsk, C, IND)$, we say the adversary win this game.

Remark: Lossiness is the property of standard lossy encryption. CPA security against outside attackers and injective inside users also remains from original ABO LPKE. But CPA security against the lossy user (we call it mirrored CPA security because the adversary knows lossy branch now) is not a natural result. Branch hidden property is implied by incapability of encryption. But it is more clear to state it formally which can help the reader to understand the difference from original version. In classic ABO lossy encryption, lossy branch is hidden to adversary and evaluation branch is public. Here the situation is reversed.

3.2. Construction

To realize a mirrored ABO lossy encryption scheme, it is required to assign every user a branch so that the user which is designated to be the one who can not generate valid ciphertext can be attached with lossy branch. A convenient approach to assign every user a branch is using key exchange protocol to obtain a shared key between receiver and every user as the user's branch. And by the property of key exchange protocol, the information of branch is hidden to other users except receiver and user themselves. This can help to inherit the CPA security against outside attackers from original lossy encryption scheme. The public key for encryption is generated after key exchange setup so that the receiver can assign one shared key as lossy branch.

We give a construction of mirrored ABO lossy encryption from a NIKE scheme $(Setup, Keygen, Sharedkey)$, a collision resistent hash function H and a standard ABO LPKE scheme (G, E, D) as follows.

- PP: Invoke $Setup$ and to generate public parameters.
- $Bgen$: Each party with ID_i runs $Keygen$ to generate (Spk_i, Ssk_i). Denote the receiver's key pair as (Spk_R, Ssk_R) and the adversary's key pair as $(Spk_{\mathcal{A}}, Ssk_{\mathcal{A}})$. Set $IND_i = (ID_i, Spk_i)$. Then $b_i = H(Sharedkey(Ssk_i, Spk_R))$.
- $Brec(IND)$: For the sender with index $IND_i = (ID_i, Spk_i)$, compute $b_i = H(Sharedkey(Spk_i, Ssk_R))$.
- $Gen(\lambda, b^*)$: Set $b^* = H(Sharedkey(Spk_{\mathcal{A}}, Ssk_R))$, and run G to obtain (s, td). Let $(lpk = s, lsk = td)$.
- $Enc(lpk, IND_i, m)$: Every user with index IND_i uses corresponding b_i and message m to invoke $E(lpk, m, b_i)$. The output is (C, ID_i) (Spk_i in IND can be omitted).
- $Dec(lsk, C, IND_i)$: Given (C, ID_i), first run $Brec(IND_i)$ to compute b_i. Then invoke $D(lsk, C, b_i)$ to obtain m.

3.3. Security

Theorem 1. *Given a NIKE scheme $(Setup, Keygen, Sharedkey)$ secure in CKS model, a collision resistant hash function H and a standard ABO LPKE scheme (G, E, D), the construction above is CPA secure against an adversary which is designated with lossy branch b^*.*

Proof. Let $\mathcal{A}$ be an adversary to attack CPA security. We build an algorithm $\mathcal{B}$ which plays the challenger for $\mathcal{A}$ to attack the security of NIKE scheme in CKS model as follows.

- Setup: On input parameters from NIKE challenger, $\mathcal{B}$ issues a honest user register query with a extract query to obtain (Spk_R, Ssk_R) and sends Spk_R with parameters to $\mathcal{A}$. Adversary supplies $Spk_{\mathcal{A}}$ to $\mathcal{B}$. Then $\mathcal{B}$ issues a corrupt user register query $(\mathcal{A}, Spk_{\mathcal{A}})$ and a reveal query $(\mathcal{A}, \mathcal{B})$ to obtain b^*. Finally, $\mathcal{B}$ invokes $Gen(\lambda, b^*)$ to generate (pk, sk) and sends pk to $\mathcal{A}$.
- Query: $\mathcal{A}$ submits (m, ID) as the query for valid encryption. $\mathcal{B}$ issues a honest user register query for ID with a reveal query $(ID, \mathcal{B})$ to obtain shared key b_{ID}. Then it computes $(C, ID) = Enc(pk, IND, m)$ which is sent to $\mathcal{A}$.
- Challenge: $\mathcal{A}$ submit two messages (m_0, m_1, ID^*) to $\mathcal{B}$. $\mathcal{B}$ issues a test query $(\mathcal{B}, ID^*)$ to obtain a challenge key b^* which is associated with ID^*. $\mathcal{B}$ flips a coin $r \in \{0, 1\}$, and computes the challenge ciphertext $(C^*, ID^*) = Enc(pk, IND^*, m_{b^*})$ and sends it to $\mathcal{A}$. $\mathcal{A}$ outputs a guess r'. If $\Pr[r' = r] - 1/2 > negl(\lambda)$, $\mathcal{B}$ outputs 1; Otherwise it outputs 0.
  If b^* is computed from a random generated key then the challenge ciphertext is random and $\mathcal{A}$ can just guess with no advantage. So we can conclude that if $\mathcal{A}$ can win the game in non-negligible advantage, $\mathcal{B}$ can distinguish b^* from random keys. This completes the reduction. □

Theorem 2. *Given a NIKE scheme $(Setup, Keygen, Sharedkey)$ secure in CKS model, a collision resistent hash function H and a standard ABO LPKE scheme (G, E, D), an adversary which is designated with lossy branch b^* can generate a valid encryption of the scheme above only with negligible probability.*

Proof. Let $\mathcal{A}$ be an adversary to attack incapability of encryption property. We build an algorithm $\mathcal{B}$ which plays the challenger for $\mathcal{A}$ to attack the security of NIKE scheme in CKS model as follows.

- Init: On input parameters from NIKE challenger, $\mathcal{B}$ issues a honest user register query with a extract query to obtain (Spk_R, Ssk_R) and sends Spk_R with parameters to $\mathcal{A}$. Adversary supplies $Spk_{\mathcal{A}}$ to $\mathcal{B}$. Then $\mathcal{B}$ issues a corrupt user register query $(\mathcal{A}, Spk_{\mathcal{A}})$ and a reveal query $(\mathcal{A}, \mathcal{B})$ to obtain b^*. Finally, $\mathcal{B}$ invokes $Gen(\lambda, b^*)$ to generate (pk, sk) and sends pk to $\mathcal{A}$.
- Query: $\mathcal{A}$ submits ID as the query for valid encryption. $\mathcal{B}$ issues a honest user register query for ID with a reveal query $(ID, \mathcal{B})$ to obtain shared key b_{ID}. Then it randomly chooses a message m and computes $(C, ID) = Enc(pk, IND, m)$ which is sent to $\mathcal{A}$.
- Forge: $\mathcal{A}$ outputs a tuple (m, C^*, ID^*). $\mathcal{B}$ issues a test query $(\mathcal{B}, ID^*)$ to obtain a challenge key b^* which is associated with ID^*. If $m = Dec(sk, C^*, IND^*)$, $\mathcal{B}$ outputs 1; Otherwise it outputs 0.

The reduction follows directly. If the forged cyphertext is valid then b^* is the shared key with probability 1. Otherwise b^* is randomly chosen. Thus we can conclude that if $\mathcal{A}$ can generate a valid encryption with non-negligible probability, then $\mathcal{B}$ can break the security of NIKE scheme in CKS model also with non-negligible probability. □

3.4. Concrete construction

We give a concrete construction in this section. The building blocks includes a NIKE scheme in Cash et al. (2009) and an ABO-LPKE from Paillier encryption scheme (Paillier, 1999) following the approach in Hemenway and Ostrovsky (2012).

- *PP*: On input security parameter λ, output a collision resistant hash function H_1 that maps any string to $l < \log N - 3 - \lambda$ bit string, and a cyclic group $\mathbb{G}$ with generator g and prime order p_1.
- *Bgen*: Each party ID_i chooses its secret key (x_i, y_i) and publishes public key as $(X_i = g^{x_i}, Y_i = g^{y_i})$. Every party computes a shared key with receiver ID_R that $Shardkey_i = (g^{x_1 x_R}, g^{x_1 y_R}, g^{y_1 x_R}, g^{y_1 y_R})$.
- *Gen*(λ, b^*): On input security parameter λ, receiver outputs a composite $N = pq$ where p and q are large primes with same length as well as a hash function H_2 which maps any string to $\mathbb{Z}_N$. Then it chooses an ID^* as a lossy identity and computes $b^* = H_2(Shardkey_i)$. Then it generates public key as $pk = (1 + N)^{b^*} r^N \bmod N^2$.
- *Brec*(*IND*): On input *IND*, output the branch as $b_i = H_2(Shardkey_i)$.
- *Enc*(*lpk*, IND_i, *m*): When ID_i encrypts an *l* bit long message *m*, it computes $C = (C_1, C_2) = ((1 + N)^{b_i} r^N / pk)^x \bmod N^2, H_1(x) \oplus m)$ and send (*C*, *ID*).
- *Dec*(*lsk*, *C*, IND_i): Receiver first recovers b_i and computes $x' = ([C_1^{\phi(N)} \bmod N^2] - 1)/N\dot{\phi}(N)^{-1} \bmod N$, $m = C_2 \oplus H_1(x')$.

3.5. Applications

This primitive can easily be embedded in a protocol with other primitives. As an example, we will modify the standard blind signature procedure that the first message is required to be sent under ABO lossy encryption. As a result, certain user is excluded to obtain valid signature while other users keep the same functionality as original. In a formal proof, the challenger can respond the query on signatures with random value for the excluded adversary.

4. SA-PEKS Secure against online keyword guessing attack (OSA-PEKS)

In this chapter, we give the approach to resist online attacks via adopting mirrored ABO lossy encryption.

4.1. Definition and security model of OSA-PEKS

In this section we define OSA-PEKS and give the model of online attack from honest registration.

An OSA-PEKS scheme is composed of 8 algorithms (*OPP, OPreset, $OKgen_{AS}$, $OKgen_R$, ODK, OPEKS, OTrapdoor, OTest*) as follows:

- *OPP*: This algorithm generates public parameters for the system by taking security parameters as input.
- *OPreset*: This algorithm runs between aided server and users including main server in the initial stage for key exchange protocol. The algorithm includes two steps: First every user generates its own key pair and then make registration on aided server. Aided server maintains a list to store information of registered users.
- $OKgen_{AS}$: On input security parameters, output parameters for aided server, including public/secret key pairs for access control.
- $OKgen_R$: On input security parameters, output parameters for main server, including public/secret key pairs for encryption functionality in PEKS. The next 4 algorithms together with this one are just the same as the definition defined in Chen et al. (2016a).
- *ODK*: This algorithm is a protocol between sender and aided server which takes keyword *w* as input to generate derived keyword odk_w.
- *OPEKS*: Generate ciphertexts which will be sent to main server to be stored for search.
- *OTrapdoor*: Compute trapdoor from derived keyword.
- *OTest*: Check whether the ciphertext is the one matching the trapdoor.

Note: There are three key generation processes in this definition. One is for key exchange protocol to establish a simple secure channel. The one run by aided server generates keys for access control to exclude online inside adversary like main server. The other run by main server is just as the same as normal key generation in PEKS schemes.

As apposed to the one more unforgeable security defined in Chen et al. (2016a), we give a more general definition to capture online attack with honest registration for the security of OSA-PEKS.

Security under online chosen keyword attack with honest registration (OCKA) is defined in the game below between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$.

- Init: $\mathcal{B}$ runs *OPP* to generate public parameters. $\mathcal{A}$ runs $OKgen_R$ to obtain his own key pairs and then register his identity with the challenger by *OPreset*.$\mathcal{B}$ runs $OKgen_{AS}$ to generate key pairs and publishes public key after $\mathcal{A}$.
- Query: The adversary can adaptively make any query on arbitrary keywords via *ODK* with the challenger for derived keywords.
- Forge: $\mathcal{A}$ outputs a pair (w, odk_w) such that $odk_w = ODK(w)$. If the adversary can forge a valid pair, we say the adversary win this game. The advantage is defined as $Adv_{OCKA} = \Pr[\mathcal{A}wins]$

If $Adv_{OCKA} < \epsilon$ where ϵ is negligible, we say the OSA-PEKS scheme is OCKA secure.

4.2. Construction of OSA-PEKS

Given a mirrored ABO LPKE scheme (*PP, Bgen, Gen, Enc, Dec*), a blind signature scheme (*BS.gen, BS.blind, BS.sign, BS.unblind, BS.ver*) and a PEKS scheme (*Kgen, PEKS, Trapdoor, Test*), we give next 8 algorithms for OSA-PEKS scheme in a framework with one aided server. The relations between primitives are shown in Fig. 2. We make LBO LPKE function on the first message of the blind signature scheme to turn it to an ABO blind signature scheme. Then combined with the PEKS scheme, we can obtain our OSA PEKS scheme.

- *OPP*(λ): Run *PP* to obtain public parameters.
- *OPreset*(λ): Every party with ID_i (including ID_{AS} and ID_{MS}) runs *Bgen* to generate (Spk_i, Ssk_i). Every party registers the public key on aided server and the aided server maintains the list.
- $OKgen_{AS}(\lambda)$: Aided server computes $b^* = Brec(IND_{MS})$ first. Then it runs *Gen*(λ, b^*) to obtain public/secret key pairs and (lpk_{AS}, lsk_{AS}). It also runs *BS.gen* to generate signature key pair (*vk*, *sk*).
- $OKgen_R(\lambda)$: Just the same as *Kgen*. The main server generates (pk_R, sk_R).
- *ODK*(w, pk_{AS}, sk_{AS}): Every sender computes $c_w = Enc(lpk_{AS}, IND_R, BS.blind(w))$ and sends (c_w, ID_S) to the aided server. The aided server computes $Dec(lsk_{AS}, IND_S, c_w)$ to obtain $BS.blind(w)$. Then it signs the blinded message with $s' = BS.sign(BS.blind(w))$. Receiving the blinded signature the sender computes $odk_w = BS.unblind(s')$.
- *OPEKS*(pk_R, dk_w): Same as *PEKS*(pk_R, odk_w).
- *OTrapdoor*(sk_R, dk_w): Same as *Trapdoor*(sk_R, dk_w).
- *OTest*(CT_{dk_w}, T_{dk_w}): Same as *Test*(CT_{dk_w}, T_{dk_w}).

The concrete communication between sender/receiver and aided server is shown in Table 2. Here we give a sketch of interactions between different roles in the scheme. The whole process is constituted by 3 stages. In the initial stage, main server generates key pair for PEKS via *OPP*(λ). Every party generates key pair for key exchange via *OPreset*(λ). And Aided server generates keys for ABO blind signature scheme via $OKgen_{AS}(\lambda)$. In the second stage, every client (including sender and user) runs ABO blind signature protocol with aided server to get derived keyword. In this stage, main

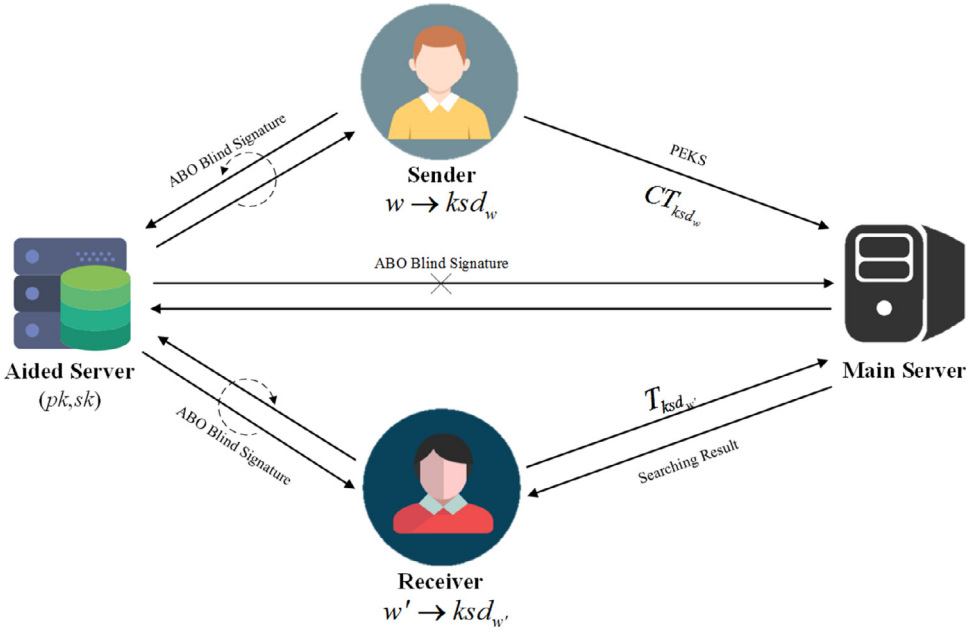

Fig. 1. The system model.

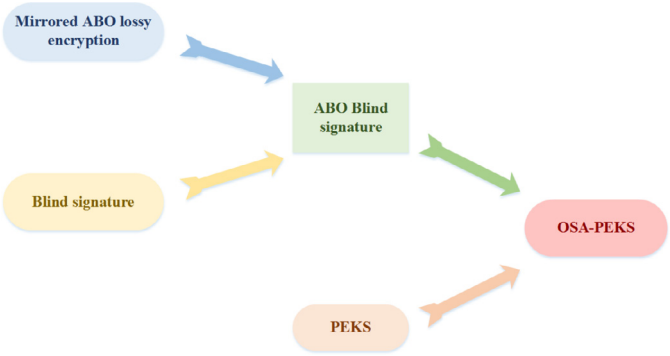

Fig. 2. Relations between primitives.

Table 2
ABO blind signature.

Sender/receiver		Aided server
$OPreset(\lambda) \to (Spk_i, Ssk_i, b_i)$		$OPreset(\lambda) \to (Spk_{AS}, Ssk_{AS})$
		$OKgen_{AS}(\lambda) \to (b^*, lpk_{AS}, lsk_{AS}, vk, sk)$
$c_w = Enc(lpk_{AS}, IND_R, BS.blind(w))$		
	$\xrightarrow{c_w}$	
		$BS.blind(w) \leftarrow Dec(lsk_{AS}, IND_S, c_w)$
		$s' = BS.sign(BS.blind(w))$
	$\xleftarrow{s'}$	
$odk_w = BS.unblind(s')$		

server can also query signature from aided server but it won't get a valid one. In the final stage, clients use the derived keyword to make trapdoors and send trapdoors to main server for search results.

4.3. Security

Theorem 3. *The construction above is CKA secure against online keyword guessing attacks assuming no dummy users registration and the properties of underlying ABO lossy encryption, blind signature scheme and PEKS schemes.*

Proof. Sketch: The proof proceeds by constructing a challenger $\mathcal{B}$ as an adversary attacking normal PEKS scheme, invoking an adversary $\mathcal{A}$ attacking our scheme as internal procedure. If the success of $\mathcal{A}$ can lead to the success of $\mathcal{B}$ with non-negligible probability, we make a reduction that our scheme is OCKA secure as if the underlying PEKS scheme is secure. We give a sketch here because we would like to highlight the difference between existing proofs and ours.

In the proof, $\mathcal{B}$ has to simulate an environment which is indistinguishable from real one for $\mathcal{A}$. Thus $\mathcal{B}$ can transform its ability to attack normal PEKS to the weapon to attack our scheme. Specifically, $\mathcal{B}$ needs to answer several kinds of queries from $\mathcal{A}$. If all the answers are indistinguishable to the real ones, the simulation succeeds. The only difference from existing proofs is that in our scheme adversary is allowed to get access to aided server now for blind signature during the derived keyword query procedure. The attacker can ask for a signature on any keyword just like normal sender to aided server. To simulate the answer, the challenger can simply respond with random value because of security of ABO lossy encryption (Theorem 2). Any way, the responce the adversary obtains is not a valid signature. So the adversary are not able to distinguish the right response with random one. Combing with incapability of encryption property, we can conclude that the adversary gains no advantage from the added oracle. Thus the security remains. □

4.4. Discussions

In practice, more adversarial behaviors still exist. If the main server is allowed to register valid dummy users or corrupt existing users, the keyword guessing attack can be easily launched even there is an identity checking process. But we can make a remedy if the corruption is detected. We can treat the set of lossy branch as a black list. LTFs which contains more lossy branches are already existing. So if we use ABN or ABM LTFs to build ABN or ABM lossy encryption scheme, we can build blind signature schemes which do not allow multiple users to get signatures. By using this extended version of lossy encryption scheme, further attack will be stopped after detection by setting the shared key as the lossy branch.

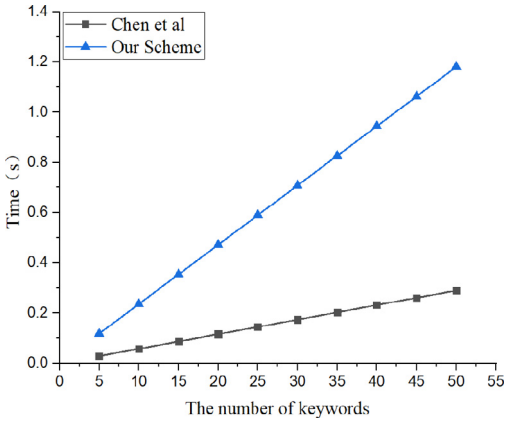

Fig. 3. The time cost of PEKS generation algorithm.

5. Implementation

To make comparison we choose the same PEKS scheme and blind signature scheme as (Chen et al., 2016a) combining with our ABO LPKE scheme in Section 3.

- $OPP(\lambda)$: On input security parameter, outputs include prime order cyclic groups $(\mathbb{G}, \mathbb{G}_1, \mathbb{G}_T)$ with generator (g, g_1, g_T), where $|\mathbb{G}| = p$, $|\mathbb{G}_1| = |\mathbb{G}_T| = p_1$. There exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. Outputs also include a composite which is the multiple of two large prime $N_s = p_s q_s$ as well as collision resistent hash functions H_1 that maps any string to $\log|N_s|$ bits string and H that maps arbitrary string to $\mathbb{G}_1$.
- $OPreset(\lambda)$: Each party ID_i chooses its secret key (x_i, y_i) and publishes public key as $(X_i = g^{x_i}, Y_i = g^{y_i})$. Every party computes a shared key with receiver ID_R that $Shardkey_i = (ID_i, g^{x_1 x_R}, g^{x_1 y_R}, g^{y_1 x_R}, g^{y_1 y_R})$.
- $OKgen_{AS}(\lambda)$: On input security parameter λ, aided server outputs a composite $N_e = p_e q_e$ where p_e and q_e are large primes with the same length as well as a hash function H_2 which maps any string to $\mathbb{Z}_N$. Then it sets the main server ID_{MS} as a lossy identity and computes $b_{MS} = H_2(Shardkey_{MS})$. Then it generates public key as $pk = (1+N)^{b^*} r^N \bmod N^2$ where $r \in_R \mathbb{Z}_N^*$. For signature key pair, it chooses a verification key $\hat{e}$ and a signing key d that $\hat{e}d \equiv 1 \bmod N_s$.
- $OKgen_R(\lambda)$: Randomly choose an α and computes $h = g_1^{\alpha}$. h is the public key for PEKS.
- $ODK(w, pk_{AS}, sk_{AS})$: Given a keyword $w \in \mathbb{Z}_{N_s}^*$, sender first blinded by choosing $r_1 \leftarrow \mathbb{Z}_{N_s}^*$ to compute $w' = r_1^{\hat{e}} w$. Then it computes $C_{w'} = (C_1, C_2) = (((1+N)^{b_i} r_1^N / pk)^x \bmod N^2, H_1(x) \oplus w')$ and sends it to aided server. Aided server first decrypt to obtain w' and outputs a blind signature $(w', s' = r_1 w^d)$. The derived key for PEKS is computed as $odk_w = H(w, r_1^{-1} s')$.
- $OPEKS(pk_R, dk_w)$: With odk_w, the sender picks random $r_2 \leftarrow \mathbb{Z}_{p_1}^*$ and computes $CT = (g_1^{r_2}, t = e(odk_w, h^r))$.
- $OTrapdoor(sk_R, dk_w)$: With odk_w, receiver computes $T = odk_w^{\alpha}$.
- $OTest(CT_{dk_w}, T_{dk_w})$: With $CT = (A, B)$, the main server checks if $e(A, T) = B$. If it is true, return 1. Otherwise, return 0.

5.1. Evaluation

In this section, We implement our scheme to evaluate its efficiency, which is based on JPBC 2.0.0 library[3] and coding language Java. We select Type A pairings are constructed on the curve $y^2 = x^3 + x$ over the field $\mathbb{F}_q$ for some prime $q = 3 \mod 4$. The process

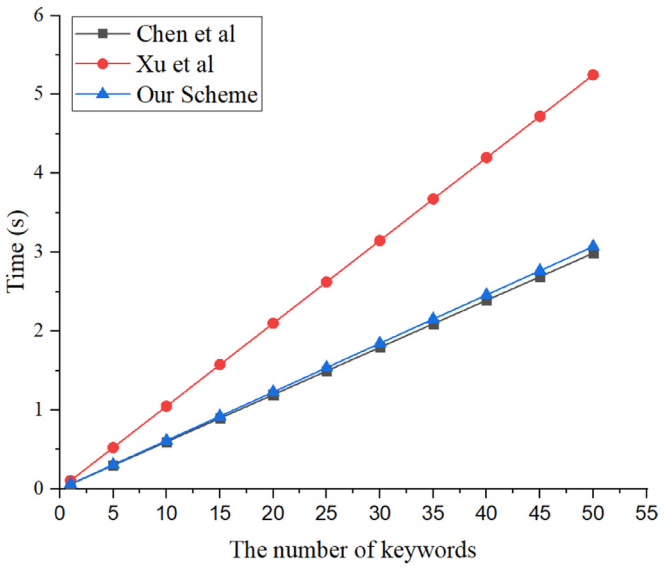

Fig. 4. The time cost of PEKS algorithm.

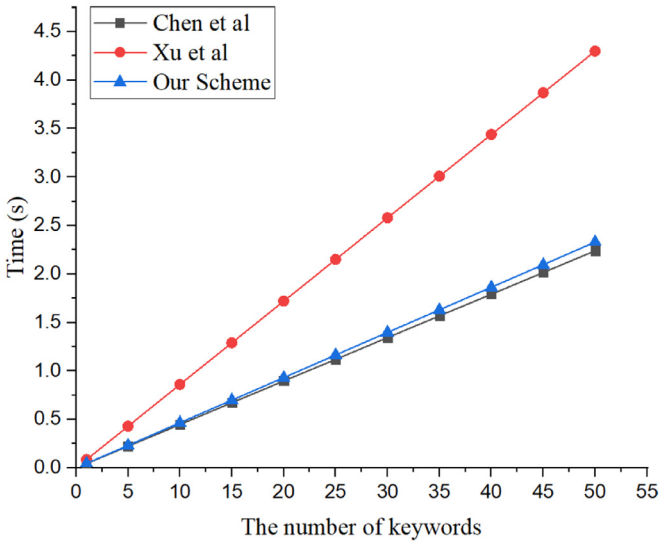

Fig. 5. The time cost of trapdoor generation algorithm.

of obtaining the random numbers is based on the class function of JPBC library[4]. We process and performance the random numbers based on a random function[5]. The function uniform randomly chooses parameters. The following experiments are based on Dell laptop (Windows 7 operation system with Intel(R) Core(TM) i5-2450M CPU 2.50 GHz, 4.00GB RAM and 500G disk storage). We used Enron email dataset which was collected and prepared by the CALO Project (A Cognitive Assistant that Learns and Organizes). It contains data from about 150 users, mostly senior management of Enron with newer version of the dataset on May 7, 2015 Version. We randomly select 64 documents as the test data. The Type of files be used to encryption is TXT format documents. We analyze and compare our scheme with Chen et al.'s scheme and Xu et al.'s scheme based on the real-world dataset from enron email dataset with the number of documents, distinct keywords and distinct keyword-document pairs are 2^6, 2^6 and 2^8, respectively. We evaluate the time cost of the blind signaure algorithm, the PEKS

[3] http://gas.dia.unisa.it/projects/jpbc/index.html#.VTDrLSOl_Cw.

[4] http://gas.dia.unisa.it/projects/jpbc/java-docs/api/index.html.

[5] http://gas.dia.unisa.it/projects/jpbc/docs/pairing.html.

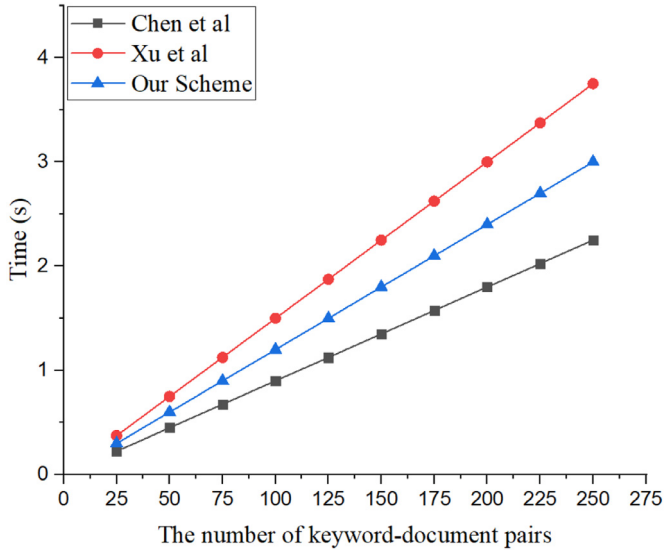

Fig. 6. The time cost of search.

algorithm, the trapdoor generation algorithm and the time cost of search. We evaluate the time cost of the algorithms by using the average time cost by 1000 rounds.

As shown in Fig. 3, in the blind signature algorithm, the time cost of our scheme is obviously more than Chen et al. scheme because we make our ABO lossy encryption scheme function on the first message in normal blind signature scheme. More precise, the computation time for 50 keywords is 0.29s for Chen et al.'s scheme and our scheme is 1.18s. There is no blinding process in Xu et al.'s scheme. (Figs. 4–6)

The time cost of the PEKS algorithm and the trapdoor generation algorithm are shown in.

CRediT authorship contribution statement

Yi Zhao: Conceptualization, Methodology, Formal analysis, Writing - original draft, Visualization. **Jianting Ning:** Conceptualization, Validation, Writing - review & editing. **Kaitai Liang:** Conceptualization, Writing - original draft, Writing - review & editing, Investigation. **Yanqi Zhao:** Software, Resources, Data curation. **Liqun Chen:** Supervision, Validation, Writing - review & editing, Project administration. **Bo Yang:** Supervision, Validation, Writing - review & editing, Funding acquisition.

Acknowledgment

This work is supported by the National Key R&D Program of China (2017YFB0802000), the National Natural Science Foundation of China (61772326, 61802242, 61802241, 61972094), the Fundamental Research Funds for the Central Universities, CHD (300102240102), the Natural Science Basic Research Plan in Shaanxi Province of China (2018JQ6088), the National Cryptography Development Fund during the 13th Five-year Plan Period (MMJJ20180217).

References

Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H., 2008. Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. J. Cryptol. 21 (3), 350–391. doi:10.1007/s00145-007-9006-6.

Bellare, M., Boldyreva, A., O'Neill, A., 2007. Deterministic and efficiently searchable encryption. In: Menezes, A. (Ed.), Proceedings of the 27th Annual International Cryptology Conference, Santa Barbara, CA, USA. Springer, pp. 535–552. doi:10.1007/978-3-540-74143-5_30.

Bellare, M., Hofheinz, D., Yilek, S., 2009. Possibility and impossibility results for encryption and commitment secure under selective opening. In: Joux, A. (Ed.), Proceedings of the 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany. Springer, pp. 1–35. doi:10.1007/978-3-642-01001-9_1.

Boneh, D., Crescenzo, G.D., Ostrovsky, R., Persiano, G., 2004. Public key encryption with keyword search. In: Cachin, C., Camenisch, J. (Eds.), Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland. Springer, pp. 506–522. doi:10.1007/978-3-540-24676-3_30.

Byun, J.W., Rhee, H.S., Park, H., Lee, D.H., 2006. Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In: Jonker, W., Petkovic, M. (Eds.), Proceedings of the Third VLDB Workshop on Secure Data Management, SDM 2006, Seoul, Korea. Springer, pp. 75–83. doi:10.1007/11844662_6.

Cash, D., Kiltz, E., Shoup, V., 2009. The twin Diffie-Hellman problem and applications. J. Cryptol. 22 (4), 470–504. doi:10.1007/s00145-009-9041-6.

Chen, R., Mu, Y., Yang, G., Guo, F., Huang, X., Wang, X., Wang, Y., 2016a. Server-aided public key encryption with keyword search. IEEE Trans. Inf. Forens. Secur. 11 (12), 2833–2842. doi:10.1109/TIFS.2016.2599293.

Chen, R., Mu, Y., Yang, G., Guo, F., Wang, X., 2016b. Dual-server public-key encryption with keyword search for secure cloud storage. IEEE Trans. Inf. Forens. Secur. 11 (4), 789–798. doi:10.1109/TIFS.2015.2510822.

Du, M., Wang, Q., He, M., Weng, J., 2018. Privacy-preserving indexing and query processing for secure dynamic cloud storage. IEEE Trans. Inf. Forens. Secur. 13 (9), 2320–2332.

Fang, L., Susilo, W., Ge, C., Wang, J., 2013. Public key encryption with keyword search secure against keyword guessing attacks without random oracle. Inf. Sci. 238, 221–241. doi:10.1016/j.ins.2013.03.008.

Gritti, C., Susilo, W., Plantard, T., Liang, K., Wong, D.S., 2014. Empowering personal health records with cloud computing: how to encrypt with forthcoming fine-grained policies efficiently. JoWUA 5 (4), 3–28. http://isyou.info/jowua/papers/jowua-v5n4-1.pdf

Han, J., Yang, Y., Liu, J.K., Li, J., Liang, K., Shen, J., 2018. Expressive attribute-based keyword search with constant-size ciphertext. Soft Comput. 22 (15), 5163–5177. doi:10.1007/s00500-017-2701-9.

He, D., Ma, M., Zeadally, S., Kumar, N., Liang, K., 2018. Certificateless public key authenticated encryption with keyword search for industrial internet of things. IEEE Trans. Ind. Informat. 14 (8), 3618–3627. doi:10.1109/TII.2017.2771382.

Hemenway, B., Libert, B., Ostrovsky, R., Vergnaud, D., 2011. Lossy encryption: constructions from general assumptions and efficient selective opening chosen ciphertext security. In: Lee, D.H., Wang, X. (Eds.), Proceedings of the 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea. Springer, pp. 70–88. doi:10.1007/978-3-642-25385-0_4.

Hemenway, B., Ostrovsky, R., 2012. On homomorphic encryption and chosen-ciphertext security. In: Fischlin, M., Buchmann, J.A., Manulis, M. (Eds.), Proceedings of the 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany. Springer, pp. 52–65. doi:10.1007/978-3-642-30057-8_4.

Hofheinz, D., 2012. All-but-many lossy trapdoor functions. In: Pointcheval, D., Johansson, T. (Eds.), Proceedings of the 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK. Springer, pp. 209–227. doi:10.1007/978-3-642-29011-4_14.

Hwang, Y.H., Lee, P.J., 2007. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (Eds.), Proceedings of the First International Conference on Pairing-Based Cryptography - Pairing, Tokyo, Japan. Springer, pp. 2–22. doi:10.1007/978-3-540-73489-5_2.

Liang, K., Huang, X., Guo, F., Liu, J.K., 2016a. Privacy-preserving and regular language search over encrypted cloud data. IEEE Trans. Inf. Forens. Secur. 11 (10), 2365–2376. doi:10.1109/TIFS.2016.2581316.

Liang, K., Su, C., Chen, J., Liu, J.K., 2016b. Efficient multi-function data sharing and searching mechanism for cloud-based encrypted data. In: Chen, X., Wang, X., Huang, X. (Eds.), Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2016, Xi'an, China. ACM, pp. 83–94. doi:10.1145/2897845.2897865.

Liang, K., Susilo, W., 2015. Searchable attribute-based mechanism with efficient data sharing for secure cloud storage. IEEE Trans. Inf. Forens. Secur. 10 (9), 1981–1992. doi:10.1109/TIFS.2015.2442215.

Liu, J.K., Au, M.H., Susilo, W., Liang, K., Lu, R., Srinivasan, B., 2015. Secure sharing and searching for real-time video data in mobile cloud. IEEE Netw. 29 (2), 46–50. doi:10.1109/MNET.2015.7064902.

Meng, R., Zhou, Y., Ning, J., Liang, K., Han, J., Susilo, W., 2017. An efficient key-policy attribute-based searchable encryption in prime-order groups. In: Okamoto, T., Yu, Y., Au, M.H., Li, Y. (Eds.), Proceedings of the 11th International Conference, ProvSec 2017, Xi'an, China. Springer, pp. 39–56. doi:10.1007/978-3-319-68637-0_3.

Ning, J., Xu, J., Liang, K., Zhang, F., Chang, E.-C., 2019. Passive attacks against searchable encryption. IEEE Trans. Inf. Forens Secur. 14 (3), 789–802.

Paillier, P., 1999. Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (Ed.), Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic. Springer, pp. 223–238. doi:10.1007/3-540-48910-X_16.

Peikert, C., Vaikuntanathan, V., Waters, B., 2008. A framework for efficient and composable oblivious transfer. In: Wagner, D.A. (Ed.), Proceedings of the 28th

Annual International Cryptology Conference, Santa Barbara, CA, USA. Springer, pp. 554–571. doi:10.1007/978-3-540-85174-5_31.
Peikert, C., Waters, B., 2011. Lossy trapdoor functions and their applications. SIAM J. Comput. 40 (6), 1803–1844. doi:10.1137/080733954.
Rhee, H.S., Susilo, W., Kim, H.-J., 2009. Secure searchable public key encryption scheme against keyword guessing attacks. IEICE Electron. Express 6 (5), 237–243. doi:10.1587/elex.6.237.
Rivest, R.L., Shamir, A., Adleman, L.M., 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21 (2), 120–126. doi:10.1145/359340.359342.
Song, D.X., Wagner, D.A., Perrig, A., 2000. Practical techniques for searches on encrypted data. In: Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, California, USA. IEEE Computer Society, pp. 44–55. doi:10.1109/SECPRI.2000.848445.
Wang, P., Wang, H., Pieprzyk, J., 2007. Common secure index for conjunctive keyword-based retrieval over encrypted data. In: Jonker, W., Petkovic, M. (Eds.), Proceedings of the 4th VLDB Workshop on Secure Data Management, SDM 2007, Vienna, Austria. Springer, pp. 108–123. doi:10.1007/978-3-540-75248-6_8.
Xu, P., Jin, H., Wu, Q., Wang, W., 2013. Public-key encryption with fuzzy keyword search: a provably secure scheme under keyword guessing attack. IEEE Trans. Comput. 62 (11), 2266–2277. doi:10.1109/TC.2012.215.

Yi Zhao received the Ph.D. degree from the School of Computer Science, Shaanxi Normal University, Xi'an, China. He is currently a lecturer at School of Information Engineering, Chang'an University, Xi'an, China. His research interests include information security and cryptography.

Jianting Ning received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University in 2016. He is currently a research fellow at Department of Computer Science, National University of Singapore. His research interests include applied cryptography and information security, in particular, Public Key Encryption, Attribute-Based Encryption, and Secure Multi-party Computation.

Kaitai Liang received the Ph.D. degree from the Department of Computer Science, City University of Hong Kong. He is currently an Assistant Professor with the Department of Computer Science, University of Surrey, U.K. His research interests include applied cryptography and information security, in particular, encryption, blockchain, post-quantum crypto, privacy enhancing technology, and security in cloud computing.

Yanqi Zhao is currently a Ph.D. candidate of School of Computer Science, Shaanxi Normal University, Xi'an, China. His research interests are digital signatures and blockchain.

Liqun Chen is a professor at the University of Surrey. Prior to this appointment in 2016, she was a principal research scientist at Hewlett-Packard Laboratories, which she joined in 1997. She has developed several cryptographic schemes adopted by the International Standards and some of them have been implemented in Trusted Platform Modules. She has an extensive publication record and holds a large number of granted patents in cryptography and information security. She has served on the editorial board for 4 international journals, as the PC (co-)chair for 14 international conferences and as the (co)-editor for 7 ISO/IEC standard documents.

Bo Yang received the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 1999. He is currently a Professor with the School of Computer Science, Shaanxi Normal University, Xi'an, China. His research interests include information security and cryptography.